Distributed Security Architectures

Fourth Quarter 2001 Progress Report

Covers work done July through September, 2001

Personnel:

Staff: Mary Thompson, Abdelilah Essiari, Srilekha Mudumbai

Students: Willie Chin, Maria Kulik, Oleg Kolesnikov, Guillaume Farret

New Proxy certificate standard

Extensions to IETF standards to support X.509 Proxy Certificates are being pursued at the IETF meetings. The proposed proxy certificate profile is a cleaned up version of the proxy certificates implemented by GSI. The TLS extension formalizes a protocol for the creation of such certificates. We investigated the new TLS record protocol to create the proposed proxy certificates and implemented code in the GSS/GSI library to create the new style proxy certificates. We now think that the proposed TLS protocol to initiate such proxy creation is too specialized to be acceptable by the IETF and the creation and verifying of these proxies will be done at the GSI library level.

The implementation provides code for requesting/checking/signing the new style X.509v3 proxy certificate s based on the X.509v3 Certificate Profile IETF draft and following the conventions used in the Globus+OpenSSL code. The implementation was separated into several logical components so that the code could be easily used in further development to incorporate delegation of credentials over TLS. The first implemented component is a fully-functional PKCS#10 certificate request generator that produces requests with embedded proxy certificate information. Although additional information is passed, the proxy certificate request remains compatible with X.509v3 specs. The second component is a validator, performing basic checks of certificate requests prior to signing them. The nature and type of the checks are based on what was described in the IETF draft. The third component signs a certificate request and returns a new proxy certificate. The component supports creation of a new proxy certificate either based on an end-entity certificate or on another proxy certificate, verifying that it adheres to the rules described in the IETF proxy certificate draft (e.g. making sure that the base certificate is either end-entity certificate or proxy certificate, checking subj/subjAltName/issuerAltName etc.) The code also implements an ASN.1 ProxyCertInfo extension, inheriting subj/issuerAltName etc. Currently, a test {id-ce 127}ASN.1 extension/objectId is used (as there it is still undefined in the draft) until an official Id is reserved for proxy certificate extension. The fourth component verifies a proxy certificate/chain of certificates. While the current version of the implementation is working, it is still a very deep beta and is undergoing a switch to incorporate the new functionality introduced in the August 2001 version of the draft. Among the new things that have already been added are ProxyCertInfo extension, KeyUsage, hashes, and checks.

Certificate Generators

An XML DTD definition of the Akenti certificates was written. It can be found at http://www-itg.lbl.gov/Akenti/docs/AkentiCertDTD.txt. A command line program was written that would take an XML description of a certificate, put iit n canonical form and sign it to create a Akenti certificate. Progress was made on adding a GUI certificate generator for the Policy certificate as well.

Apache-Akenti server

An authorization module to enforce Akenti access policy was written for use with Apache 1.3.This module allows resources accessed via a Web server to use the same Akenti policy as resources accessed by other means. It allows for a rich set of policies based on the Web client identifying himself with an X.509 identity certificate.

Standalone Server

The Akenti policy engine provides a function call interface to discover a user's access rights for a resource. The Standalone Akenti server provides a message interface for the same purpose in order to facilitate the use of Akenti from languages other than C++ or C. Using the server also allows the Akenti server to be run on a different machine than the resource gatekeeper. The server supports requests to find out the access rights of a user to a resource, and a request for the Akenti server to flush any cached certificates. The access request takes the name of an Akenti principal (DN and CA), the name of a resource and returns a capability certificate including the principal, resource and the set of rights that the principal has.

During this quarter a simple message protocol for this server was designed and server and client side libraries that implement this protocol were written. This code was designed so that the protocol can be extended and multiple protocols can be supported.

Supporting servers

There is a caching server written in Java that the Akenti policy engine can use to cache the certificates that it has gathered in making a policy decision. It is also used to cache policy decisions in the form of capability certificates. During this quarter this server was greatly improved. It now uses structured messages and a more efficient caching scheme. It implements a new interface to allow requests to flush cached certificates.

The Logging server was also improved and tested. In particular considerable effort was involved in designing the structure and content of the logging messages to give a understandable description of the whole access control process. The number of logging messages is controlled at run time by parameters in the Akenti configuration file. At a standard monitoring level of logging the log history should allow a stakeholder or user to see all the certificates involved in making a policy decision. The cause of any failures should be obvious from the log. Higher levels of monitoring can be turned on to get more information about failures.

Akenti Policy Engine

The Akenti Policy engine was modified to incorporate a new certificate verification scheme that takes into account the level at which certificates were verified. This new verification scheme will result in less verification of x509 and attribute certificates which can and most likely will be used at different levels of the access decision process. We anticipate that these new schemes, verification and caching, will result in faster access decisions.